



In This Issue...

[SC97 Special
Section: NASA
Centers
Collaborate on Demos](#)



[Origin2000 Update](#)

[Kutler Focuses on IPG](#)

[CFD Visualization
Library](#)

[Device Modeling
Workshop](#)

[Contiguous Improvement](#)

[Whitney Team](#)

[New Approach to Mass
Storage](#)



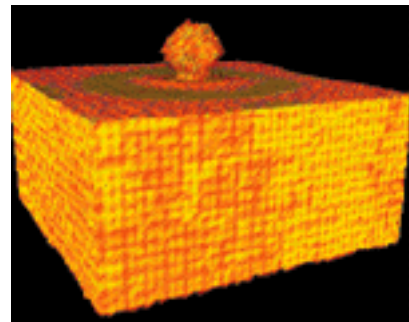
Origin2000 Update: Studies Show CFL3D Can Obtain 'Reasonable' Performance

This article describes recent performance studies done using the MPI parallel version of CFL3D on a large multi-block grid. The basic intent of these studies was to better understand and optimize the performance of CFL3D on this new system. The MPI version of CFL3D was chosen for this work partly because of its good match to the Origin2000's parallel architecture and partly because of its interest to the commercial aeronautics community.

[To The Article...](#)

Device Modeling Workshop Inspires Collaboration Among Experts From Around the World

This image: Hypervelocity impact of an 8-kilometer-per-second nanoparticle (4 nanometer diameter) on diamond coating, based on research presented by Priya Vashista at the Second NASA Device Modeling Workshop.



[To The Article...](#)



In This Issue...

[SC97 Special](#)
[Section: NASA](#)
[Centers](#)
[Collaborate on Demos](#)



[Origin2000 Update](#)

[Kutler Focuses on IPG](#)

[CFD Visualization](#)
[Library](#)

[Device Modeling](#)
[Workshop](#)

[Contiguous Improvement](#)

[Whitney Team](#)

[New Approach to Mass](#)
[Storage](#)

Origin2000 Update: Studies Show CFL3D Can Obtain 'Reasonable' Performance

by [Tom Faulkner](#)

In the July-August issue, we reported that [initial tests by Jim Taft](#) on the Silicon Graphics Inc. Origin2000 system showed promise for achieving high, sustained performance with the well-known ARC3D code. Recent performance studies by Tom Faulkner, using the MPI parallel version of another computational fluid dynamics code, CFL3D, show similar potential.

One of the first large computational fluid dynamics (CFD) programs to be ported to [turing](#), the new Silicon Graphics Inc. Origin2000 system at the NAS Facility, was CFL3D, a code developed at the Computational Fluids Laboratory, NASA Langley Research Center (LaRC). CFL3D solves the time-dependent thin-layer Navier-Stokes equations in three dimensions using a finite-volume formulation.

Within this article...

[Test Conditions](#)

[Compiler Optimization](#)
[Options](#)

[Arithmetic Precision](#)

[Floating-point](#)
[Performance](#)

[Scaling Results](#)

[Performance](#)
[Comparisons](#)

This article describes recent performance studies done using the MPI parallel version of CFL3D on a large multi-block grid. The basic intent of these studies was to better understand and optimize the performance of CFL3D on this new system. The MPI version of CFL3D was chosen for this work partly because of its good match to the Origin2000's parallel architecture and partly because of its interest to the commercial aeronautics community.

CFL3D, which utilizes block-structured grids, was parallelized by distributing the grids over a set of CPUs. A master-slave architecture was used, with the master CPU managing the flow of global data to and from the compute CPUs. The version of CFL3D used for these studies was kindly supplied by Robert Biedron of LaRC.

The problem grid used in the studies was provided by Pichuraman Sundaram ([The Boeing Co.](#), Long Beach) and represents a wing-body configuration that is part of Boeing's efforts in support of NASA's High Speed Research program. This production-quality grid contains about 3.5 million points divided into 12 blocks of dimension 25x65x181. The computation uses three multigrid levels with 1000, 500, and 100 cycles for the coarsest to finest levels. The solution has not converged completely with this set of cycles but runs long enough to provide reasonable timings on the order of one hour per run.

Test Conditions

Some of the test conditions used to do the measurements are described below. All CFL3D runs were executed with the following command line:

```
/bin/time mpirun -np <p> \
perfex -mp -e 21 cfl3d_task < wb_NS.inp
```

where <p> is the number of CPUs (master and worker), cfl3d_task is the name of the executable, and wb_NS.inp is the input file. Except for the scaling runs, all runs used 13 CPUs (1 master and 12 compute CPUs).

In the following tables, all elapsed wall-clock times are in seconds as reported by the /bin/time command. The timings include: mpirun program startup; reading and initializing data structures; periodically writing restart files; and program termination, including writing various output files.

The primary measure of floating-point performance is the average number of

floating-point instructions executed per second per CPU (Flop/sec/CPU). The actual floating-point instruction count was obtained from the hardware performance counters on the R10000 processors and provided through the perfex command. The Flop/sec/CPU value is computed as $\text{Flop/sec/CPU} = \text{Flop/sec} / (P * T)$ by dividing the floating-point instruction count by the elapsed wall-clock time and the number of compute processors.

Compiler Optimization Options

The first step was to explore the effects of some available optimizations supported by turing's Fortran compiler. The basic set of compiler options used throughout this work was:

```
-O3 -64 -mips4 -r10000 -align64 -r8 -OPT:Olimit=3000
```

Three additional compiler options were studied beyond this basic set; these are summarized in the following table.

Option	Description
PF	Compiler generated memory prefetch instructions
IEEE	IEEE arithmetic and rounding
MADD	Combined Multiply and Add instructions

Unless otherwise noted, these three options were set to: PF=ON, IEEE=OFF, and MADD=ON for all the compiler optimization runs.

The prefetch compiler option allows the compiler to insert instructions to prefetch data from memory before they are needed. This allows the compiler to load data into the R10000's caches. Then, when the data is actually needed, it can be loaded immediately from cache, avoiding the penalty of stalling the processor while accessing memory.

Unfortunately, the analysis that the compiler uses to decide where to insert prefetch instructions is based on a static analysis of the code, so the actual performance of the program may depend on a variety of run-time factors. Thus, it is always prudent to check the effect that prefetch will have on a program's performance.

PF	Wall Time	Flops	Flops/sec/proc
OFF	4977	1.67×10^{12}	28.0×10^{06}
ON	4329	1.67×10^{12}	32.1×10^{06}

Performance results of separate runs made with prefetch instructions OFF and ON are listed below.

Enabling the compiler to use the data prefetching capabilities of the R10000 processor with PF=ON improved CFL3D's overall performance by about 13 percent.

It is worth noting that further improvements in performance might be achieved by even finer control of the compiler's use of prefetch. For example, the compiler has options that allow the control of prefetch instructions between the various levels of the Origin2000 memory system.

Arithmetic Precision

The Origin2000 supports full conformance to the IEEE 754 floating-point arithmetic roundoff and overflow behavior. However, many programs do not always need this full precision and may benefit in performance by relaxing the IEEE precision requirements. This is particularly true for programs that use 8-byte floating-point data types, as is the case with CFL3D.

Another area associated with floating point precision involves certain code rearrangements that the compiler may want to do, which might cause changes in the code's floating point roundoff and overflow characteristics. The compiler allows the programmer to choose several levels of compliance in both of these areas. For simplicity, the IEEE compliance has been grouped together with the roundoff control at three levels: FULL disables all departures from IEEE arithmetic and prohibits code rearrangements that would affect roundoff; DEF uses the defaults for these two options; and OFF enables full aggressive transformations, even if they might produce inaccurate results or overflows, and relaxes the IEEE standards to the maximum extent supported by the hardware.

To gauge how these arithmetic optimizations affect CFL3D, a series of runs with OFF, DEF, and FULL optimizations were made.

IEEE	Wall Time	Flops	Flops/sec/proc
FULL	4712	1.55×10^{12}	27.4×10^6
DEF	4439	1.57×10^{12}	29.5×10^6
OFF	4329	1.67×10^{12}	32.1×10^6

The results below show a clear trend to higher performance as the requirements on IEEE precision are relaxed and code rearrangements are allowed. Overall, performance was improved about 8 percent.

To see if the precision reductions had any effect on the final results, the listing output from the DEF and FULL runs were compared to the OFF run. To the precision printed (4-5 decimal digits), no differences were found in the iteration history or the values of the final results.

Floating-point Performance

One reason the R10000 processor achieves high floating-point performance is that the CPU is capable of doing a floating-point add and multiply simultaneously. These joint multiply-add instructions (called MADD instructions) are treated the same as other floating multiply instructions. One side-effect of this is that while two floating-point operations are accomplished with a MADD instruction, the R10000 performance counters count this as a single floating-point instruction.

At higher optimization levels, the Fortran compiler uses a fair number of MADD instructions (because they improve performance). So, obtaining the floating-point performance from the floating-point counters can underestimate a program's actual floating-point performance. Fortunately, the compiler has an option to disable MADD instructions. A more accurate value of a program's floating-point performance can be obtained by using the counts from a MADD=OFF run and the elapsed wall-clock time from a MADD=ON run. Both results are shown below.

MADD	Wall Time	Flops	Flops/sec/proc
OFF	4864	2.06×10^{12}	35.2×10^6
ON	4329	1.67×10^{12}	32.1×10^6

Clearly, the fraction of MADD instructions used by the compiler is a non-trivial portion of all the floating-point instructions. For this particular dataset, it amounts to about 23 percent of the total floating-point operations.

Combining the floating-point operations count for MADD=OFF with the elapsed wall-clock time from the MADD=ON run yields a floating-point performance of $39.7 \times 10^{+06}$ Flop/sec/CPU or an aggregate floating-point performance for all 12 compute CPUs of $475 \times 10^{+06}$ Flop/sec.

Scaling Results

The final set of measurements concerns how the performance of CFL3D scales with the number of CPUs used in the calculation. The elapsed wall-clock times, floating point performance, and speedups are given below. The Speedup value for P CPU is the ratio of the 1-CPU runtime to the runtime for P CPUs.

CPUs	Wall time	Flops	Flops/sec/proc	Speedup
1	53054	$1.67 \times 10^{+12}$	$31.5 \times 10^{+06}$	1.00
2	27572	$1.67 \times 10^{+12}$	$30.3 \times 10^{+06}$	1.92
3	18565	$1.67 \times 10^{+12}$	$30.0 \times 10^{+06}$	2.86
6	8724	$1.67 \times 10^{+12}$	$31.9 \times 10^{+06}$	6.08
12	4329	$1.67 \times 10^{+12}$	$32.1 \times 10^{+06}$	12.3

One interesting aspect of these scaling results is the apparent super-linear scaling for the 6- and 12-CPU cases. The most likely explanation for this lies in the structure of turing's memory. The basic Origin2000 hardware unit is a 2-CPU node that contains a portion of memory considered local to the node.

For small programs, the memory allocation is generally local. However, for programs that need more memory than can be accommodated on a single node, its memory allocation is on other nodes, which take longer to access. As a program uses more memory, its maximum effective bandwidth to memory will begin to decrease.

As the workload is shared over fewer MPI processes, then the memory requirements for each process increases. The worst case here is the single-CPU case that uses about $1100 \times 10^{+06}$ bytes of memory. The majority of memory used is remote and, in fact, may be several "hops" across the system's interconnect network.

On the other hand, as the number of CPUs increases, the amount of memory needed by a single MPI process decreases. For the 12-CPU case, each MPI process used about $120 \times 10^{+06}$ bytes of memory--enough for the majority of

the individual processes' memory to be local. So, the superscaling results from a comparative slowdown of the 1-CPU case, relative to the performance of the 12-CPU case.

Performance Comparisons

The following presents a couple of data points for floating-point performance to compare with the results for CFL3D.

Program	Flops/sec/CPU
PEAK	$390 \times 10^{+06}$
ARC3D	$98.6 \times 10^{+06}$
CFL3D	$39.7 \times 10^{+06}$

The PEAK value is the maximum floating point performance supported by the R10000 processor. The ARC3D result is taken from Jim Taft's study (*see NAS News, July-August '97*). Taft's work represents a significant reprogramming of ARC3D's computational kernels and data structures to improve its performance, and can be considered a best-effort result for a typical CFD type of application.

As a percentage of PEAK, CFL3D's performance is 10 percent--a bit disappointing, as it is often possible to reach 50 percent of PEAK for a well-vectorized code on a Cray system. However, with a judicious choice of compiler options alone, CFL3D manages to reach about 40 percent of the floating-point performance of the best-methods optimization work on ARC3D. This should come as good news for those who want to obtain reasonably good performance on a cache-based system like turing without having to make major code revisions that are, as often as not, system-specific and non-portable.

Tom Faulkner, who conducted the CFL3D studies, has worked in the high performance computing field for nearly 20 years. He joined the NAS high speed processor support group in early 1996. Faulkner holds a Ph.D. in theoretical chemistry from the University of Minnesota.



In This Issue...

[SC97 Special](#)
[Section: NASA](#)
[Centers](#)
[Collaborate on Demos](#)



[Origin2000 Update](#)

[Kutler Focuses on IPG](#)

[CFD Visualization Library](#)

[Device Modeling](#)
[Workshop](#)

[Contiguous Improvement](#)

[Whitney Team](#)

[New Approach to Mass](#)
[Storage](#)

Kutler Focuses on Information Power Grid 'Vision'

by Ayse Sercan

Since his appointment as acting division chief of the NAS Systems Division, Paul Kutler has focused on leading the division toward a new vision: the [Information Power Grid \(IPG\)](#).



While the appointment is temporary, it's by no means a short-term position. According to Kutler, the search for a new division chief will probably take as long as a year. (For more information on the division chief job opening, see "[The Ideal Candidate](#)".) Kutler, currently deputy director of the Information Systems Directorate at Ames Research Center, replaces Marisa Chancellor, who resigned last July.

"Twenty years ago, we had a vision of a numerical wind tunnel to work towards, and we were extremely successful in accomplishing that," Kutler said. "There was, however, a need to set a new vision." Kutler, who headed the Fluid Dynamics Division when the numerical wind tunnel concept was being developed, also led the team that met late last year to discuss and create a new computing vision for Ames and NASA.

The concept the committee pursued for NASA's future is the distributed, heterogeneous computing paradigm called the Information Power Grid. As with the numerical wind tunnel, "there are enterprise users out there who, if we had a system like this, could use it right away to solve their problems," Kutler said. (NASA has four enterprises, analogous to strategic business units found in private corporations: Aeronautics, Human Space Flight, Mission to Planet Earth, and Space Science.) Beginning next year, "the Information Power Grid will, without a doubt, be the central focus for this division."

In early September, Kutler designated Cathy Schulbach -- Computational Aerosciences (CAS) project manager within the High Performance Computing and Communications (HPCC) Program -- as technical program plan developer for the IPG. She will be assisted by David DiNucci (NAS parallel tools team) and Subhash Saini (who heads NAS's newly formed modeling and simulation group), as well as individuals from other organizations at Ames and NASA's Langley and Lewis Research Centers. The IPG is planned to be an integration of the Advanced Computing, Networks, and Storage element of the Information Technology Program, managed by Tom Edwards, and the CAS element of the HPCC Program, led by Bill Feiereisen. Feiereisen is also leading the overall IPG program plan effort.

Kutler said that progress on planning the IPG is going well. "We're in the process of doing the technical program development -- in essence creating the strategic goals, milestones, requirements, partners, and advocacy for the Information Power Grid. We've briefed the NASA executive council [made up of center directors from NASA centers], and we're in the process of preparing for a program readiness review for a committee established by NASA headquarters."

After the technical program approvals have been completed, the next step is to structure the organization to execute the IPG program -- a process that may or may not require reorganization of the division, Kutler said.

Although the IPG concept is not unique -- several academic institutions and government organizations are building similar systems -- Kutler sees the IPG as an opportunity for Ames to once again take the lead in next-generation supercomputing enterprise applications. "[Ames is] a research center, and our scientists like challenges. This is an opportunity to focus our computer science program on NASA's future enterprise computing requirements." In addition, he noted that "NAS is very good at doing systems integration. We see that as a key role for us in the Information Power Grid."

[More information on the progress of the Information Power Grid project](#) is available.



In This Issue...

[SC97 Special](#)
[Section: NASA](#)
[Centers](#)



[Collaborate on Demos](#)

[Origin2000 Update](#)

Kutler Focuses on IPG

[CFD Visualization Library](#)

[Device Modeling](#)
[Workshop](#)

[Contiguous Improvement](#)

[Whitney Team](#)

[New Approach to Mass](#)
[Storage](#)

The Ideal Candidate

While acting division chief Paul Kutler guides the NAS Systems Division into preparations for the Information Power Grid, the search for a new division chief continues. "Because this is an SES [Senior Executive Service] position, it has to go through several channels," Kutler said.

The process has been intentionally delayed, and the closing date on the job announcements has been extended to January 30, 1998, to allow for advertisement in several periodicals and at several major events in the coming months. "There are some key meetings in November, such as Supercomputing 97, where we will want to make an announcement, and also the AIAA [American Institute of Aeronautics and Astronautics] meeting in January," Kutler continued.

The Executive Resources Panel will evaluate candidates based on their ability to carry out a strategic vision, as well as experience with human resources management, program development and evaluation, resource planning and management, and organizational representation and liaison. Candidates need to have strong managerial and technical skills - both in computer systems such as hardware, software, and networks, and in the needs and concerns of computational aerodynamic researchers which will help in the task of developing and operating a supercomputing facility.

The [complete job announcement](#), to be distributed to major journals through November, is available in Adobe Acrobat or Microsoft Word format.

[Return to Main Article](#)



In This Issue...

[SC97 Special
Section: NASA
Centers
Collaborate on Demos](#)



[Origin2000 Update](#)

[Kutler Focuses on IPG](#)

[CFD Visualization
Library](#)

[Device Modeling
Workshop](#)

[Contiguous Improvement](#)

[Whitney Team](#)

[New Approach to Mass
Storage](#)

CFD Visualization Library Makes New Strides in Supporting Application Development

by [Patrick Moran](#) and [Chris Henze](#)

The second generation of a library that supports the rapid development of computational fluid dynamics applications is approaching release by the NAS data analysis group. The Field Encapsulation Library (FEL), which will be released by year end, contains many new features, including added support for mesh-independent algorithm development and a powerful, new derived field capability. Derived fields allow the user to easily create new fields in terms of existing ones, just as with a calculator, and use the results as one would use any other field.

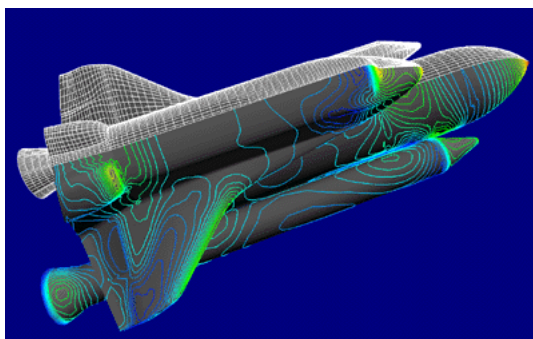
Within this article...

[Iterating Towards
Mesh Independence](#)

[The FEL Field
Family](#)

[When Laziness Is A
Virtue](#)

[Differential Operator
Fields](#)



Sample visualization of a field based on a multizone structured mesh (represented by white grid lines), using routines written in terms of the Field Encapsulation Library (FEL), which allows developers to write scientific visualization algorithms and systems in a mesh-independent manner. The isocontours (colored

lines) on the vehicle surface display the pressure field, provided through an FEL derived field. The isocontour and mesh surface routines were written in terms of FEL cell iterators. Space shuttle launch vehicle dataset provided courtesy of Pieter Buning, NASA Langley Research Center; visualization by Timothy Sandstrom and

Patrick Moran, NAS Systems Division.

[click to see 400x250 gif of image, 98K](#)

FEL is a continuation of work first introduced by Steve Bryson and David Kenwright, also in the data analysis group (*see [NAS News, December'96](#)*).

The team has used the new FEL as the foundation of a prototype system for calculating with fields, and is in the process of integrating the new version into an existing application: [the virtual windtunnel](#). It is anticipated that the library will enable the rapid development of new applications, and that the resulting software will be more easily adaptable to a wide variety of meshes and fields.

Iterating Towards Mesh Independence

FEL takes a big step toward providing a mesh-independent interface with the introduction of iterators. Iterators are an abstraction for processing a collection of items, one item at a time. For users, iterators mean easier code development and the potential for introducing new types of meshes in the future, without rewriting existing routines.

FEL provides iterator support for collections corresponding to meshes and the fields based on those meshes. Meshes can be thought of as either a set of vertices or as a set of cells. Using an iterator, a user can step through the vertices or cells of a mesh without writing code that is specific to a particular type of mesh.

For example, consider a simple visualization technique for scalar fields using a vertex (FEL_vert_pos) iterator. The following routine works with fields based on FEL structured, unstructured, or multizone meshes:

```
void vis_fun(FEL_float_field_ptr float_field) {  
  
    FEL_vert_pos_iter iter, end;  
  
    FEL_vector3f c; // coordinates
```

```
float f; //field value
```

```
float_field->begin(&iter);
```

```
float_field->end(&end);
```

```
for ( ; iter != end; ++iter) {
```

```
    float_field->coordinates_at_vert_pos(*iter, &c);
```

```
    float_field->at_vert_pos(*iter, &f);
```

```
    // ... draw using coordinates and field value f
```

```
}
```

```
}
```

Using C++ operator overloading, FEL hides the mesh-specific work behind the iterator advance (++iter), dereference (*iter), and end test (iter != end) operations.

The FEL Field Family

All fields in FEL are subclasses of a common FEL_field<T> parent class, where <T> indicates that the FEL_field class is written using C++ templates. In the case of FEL_field<T>, T specifies the node type of the field. FEL provides typedef names for the most commonly used types of scalar and vector fields (FEL_float_field and FEL_vector3f_field, respectively), so that users do not have to work with C++ template syntax.

Algorithms written in terms of the field class interface can work with instances of any field subclass, increasing user implementation reusability. For instance, the vis_fun example above could be used with any field with a float node type, including the differential operator and derived fields described below.

When Laziness Is A Virtue

The differential operator fields and derived fields in FEL both employ a technique known as "lazy evaluation." With lazy evaluation, field values are computed when they are requested, rather than in advance. For example, if a user constructed a velocity field derived from a density field and a momentum field, no velocities would be computed initially. When that user calls for a velocity at a particular location within the field, then the field executes the derivation computation, dividing momentum by density, as in the case of the velocity example.

Lazy evaluation shines in situations where an application only needs the derived values within small subregions of a field, as is typically the case when advecting streakline particles through a velocity field. The strengths of lazy evaluation grow in importance when working with large datasets in particular, time-varying datasets where precomputing derived values over the whole dataset may be impractical due to the computational time and storage requirements.

While lazy evaluation is the default, FEL can precompute the derived value at each node (so called "eager evaluation"), if preferred.

Differential Operator Fields

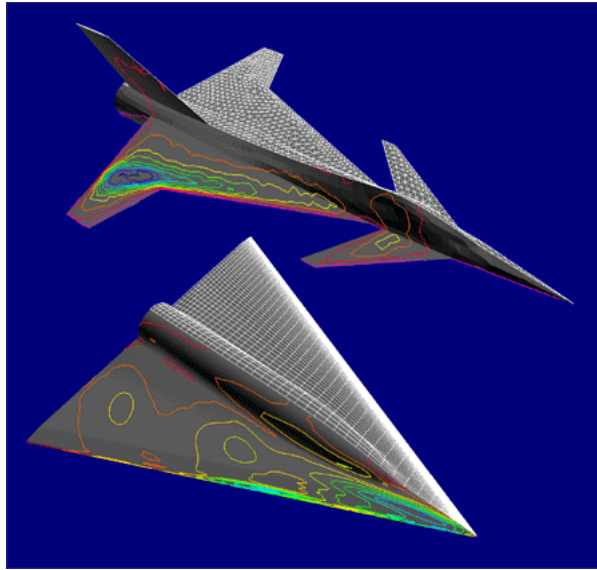
New with the upcoming release of FEL is support for the gradient, divergence, and curl differential operators. The built-in support for these commonly used operators makes it easier for users to quickly develop new applications, since more of the necessary field functionality can be used "off the shelf" from FEL. The library supports computing differential operator results by either first-order or second-order techniques, at the user's option.

General-purpose Derived Fields

Also new in the upcoming release of FEL is support for derived fields. Derived fields are a general-purpose approach to creating new fields in terms of existing fields, as in the velocity field example, above. To construct a derived field, the user provides the fields being derived from, as well as a mapping function that computes the derived value.

Derived fields may also be chained together with other derived fields or differential operator fields. For instance, a vorticity field can be

constructed using a velocity derived field and a curl differential operator field. In addition to the ability to construct arbitrary derived fields, FEL provides built-in support for some of the standard fields derived from PLOT3D solution files, such as pressure and temperature.



These images illustrate visualization of fields based on single-zone structured and unstructured meshes. As with the shuttle image (page 1), the isocontour curves are based on the pressure fields, provided by a Field Encapsulated Library (FEL) derived field. A single isocontour implementation, written in terms of FEL cell iterators, was used for all three datasets. FEL cell iterators support looping over subsets of cells in a mesh, including the

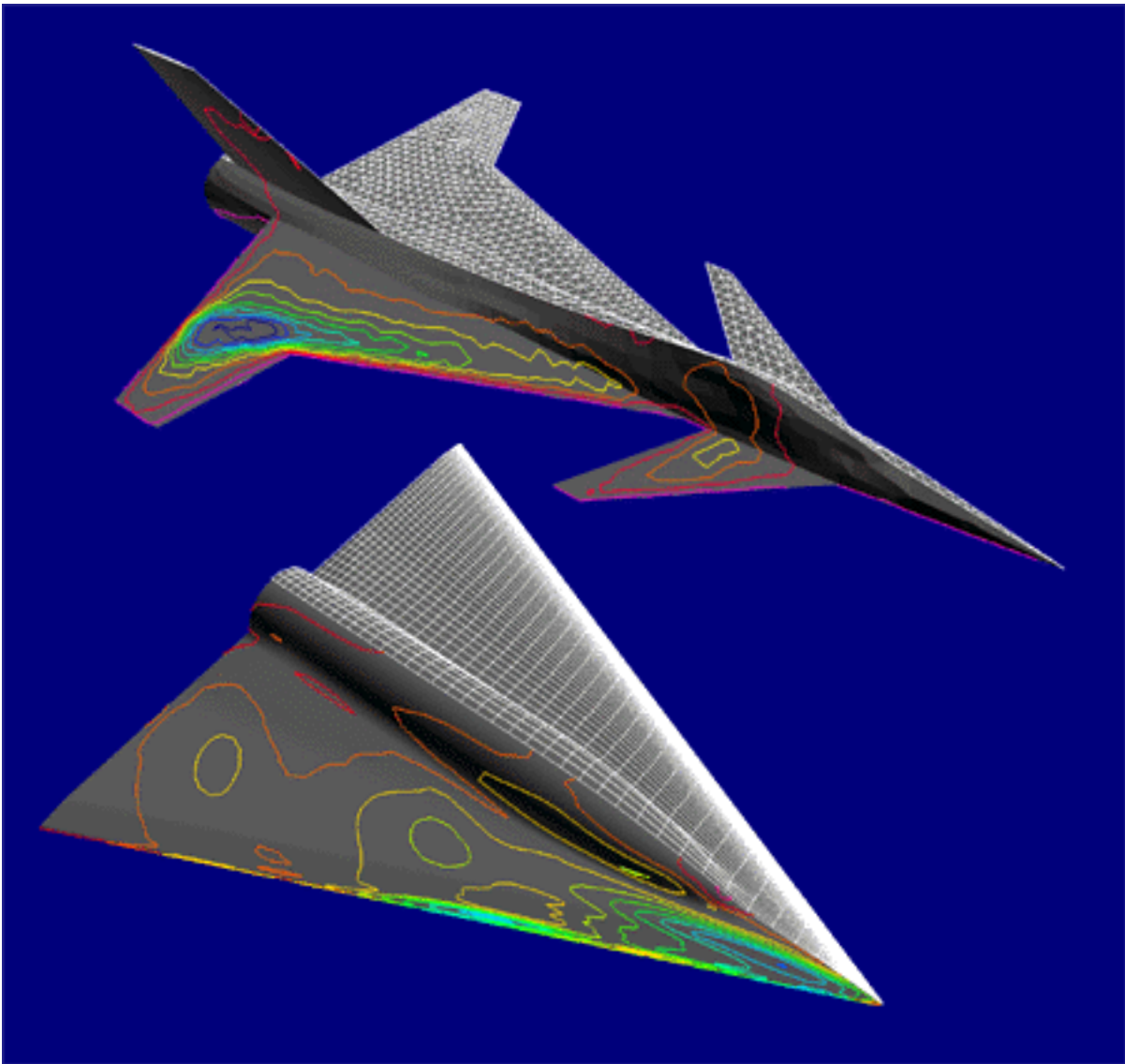
triangle or quadrilateral cells of a surface. Fighter aircraft dataset (top) by Robert Neely, Langley Research Center; delta wing dataset by Neil Chaderjian, Ames Research Center. Graphics by Timothy Sandstrom and Patrick Moran.

[click to see 450x425 gif of image, 187K](#)

For Further Information

For more information on the Field Encapsulation Library and the upcoming release, contact [Steve Bryson](#).

Patrick Moran and Chris Henze work in the [NAS data analysis group](#) and have led the technical design and development of the second-generation Field Encapsulation Library. Moran and Henze, along with Steve Bryson and David Kenwright, comprise the FEL design team.





In This Issue...

[SC97 Special](#)
[Section: NASA](#)
[Centers](#)
[Collaborate on Demos](#)



[Origin2000 Update](#)

[Kutler Focuses on IPG](#)

[CFD Visualization](#)
[Library](#)

[Device Modeling](#)
[Workshop](#)

[Contiguous Improvement](#)

[Whitney Team](#)

[New Approach to Mass](#)
[Storage](#)

Device Modeling Workshop Inspires Collaboration Among Experts From Around the World

by Ayse Sercan

The Second NASA [Device Modeling Workshop](#), held August 7 & 8 at NASA Ames Research Center and sponsored by the NAS Systems Division, brought together scientists and engineers from around the world. The workshop was successful in forging new working relationships among the 195 participants. The two-day event was characterized by an openness to sharing information and solutions to problems.

Possible Cooperation Between Ames & H-P

Ames researchers made a number of collaborative connections at the workshop, including the possibility of working with workshop speaker Stanley Williams, Hewlett-Packard (HP), Laboratories (Palo Alto, CA), who runs an experimental laboratory for HP. Williams pointed out the possible benefits for both organizations: "My group is almost entirely an experimental group. We're studying the experimental aspects of how atoms come together to form very small structures. We have produced a large amount of very detailed, quantitative data. The modeling group [at NASA] would be able to compare their simulation results with this extremely quantitative data for verification."

Within this article...

[Possible Cooperation Between Ames & H-P](#)

[NASA's Direction](#)
[With Device](#)
[Modeling](#)

[New Vision for](#)
[Information Systems](#)

[HPCC and Device](#)
[Modeling](#)

[Expertise in Process](#)
[and Device](#)
[Modeling](#)

[Device Modeling at](#)
[Ames](#)

[Keynote Echoed](#)
['Collaborative Spirit'](#)

[Budgetary Concerns](#)
[Fuel Challenges](#)

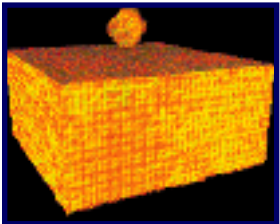
[Best Investment:](#)

[Miniature
Instruments](#)

[Future Spacecraft
Like 'Ancient
Mariners'](#)

Williams was recruited by HP after fifteen years at UCLA to form a small, long-range laboratory. Unlike other industrial labs, this one is not product-oriented. "At least 99 percent of the effort in most industrial labs is focused on what will be sold in the next three years. Long-range work is done mainly in academia and research centers," Williams explained. This product orientation keeps industry labs from doing basic research, but doesn't keep them from being interested in device modeling, he said.

Noting the high number of industry participants -- about 90 -- at the workshop, he continued, "There's quite a representation of industry here. Even though they're not able to do the work themselves, they know that within a few years' time they're going to have to be taking in these developments from the academics and the national labs and turning them into products. So they want to be prepared for that."



Hypervelocity impact of
a nanoparticle on
diamond coating

NASA's Direction With Device Modeling

Henry McDonald, Ames Research Center Director, began his talk -- which opened the workshop -- by detailing NASA's reasons for being interested in device modeling: "The top-down requirement is clearly the NASA missions that we're about to embark on and are embarking on. They provide a very stressing set of requirements for devices," he said.

McDonald noted that "one of the things that we have found is that although there is a high degree of interest beyond NASA in device physics, industry is not incentivized to respond to NASA's needs. They are highly geared to the needs of the major marketplace, and the major marketplace is *not* in providing high-performance chips to go to Mars or investigate the other planets."

Ames has a critical role in the field of device modeling, McDonald said. "Over the years, Ames has developed an expertise in solving very complex physical problems by using computational methodologies -- some very innovative." He was optimistic about applying that expertise to solving problems in condense matter physics.

New Vision for Information Systems

Paul Kutler, acting division chief of the NAS Systems Division and deputy director of Ames' Information Systems Directorate, discussed how the center will be approaching information technology in the future. As NASA's designated Center of Excellence for Information Technology (COE-IT), "We have an opportunity to form strategic partnerships and alliances with some of the organizations here, some of the universities, and also around the country, to further enhance our skills in this area," he said.

Kutler explained that part of being the COE-IT is supporting NASA missions. "The fundamental function of the Information Systems Directorate is to provide access to knowledge that enables understanding and discovery. We also provide a computational capability to [NASA], in order to allow it to carry out its missions at an economical rate."

The role of IT within NASA is more than just support, Kutler continued. "We also perform leading-edge research in the computational sciences area." For example, in addition to maintaining Ames' supercomputing facilities, the NAS Systems Division "houses a lot of the activity that is going to be talked about [at the workshop] -- the nanotechnology, the device modeling work."

HPCC and Device Modeling

William Feiereisen, who manages NASA's [High Performance Computing and Communications \(HPCC\) Program](#), discussed how this program fits in with the device modeling research at Ames. HPCC began with the push for teraflop computing within NASA. "A number of agencies banded together and started an advocacy to try and put together a large, funded program to be able to advance

computer technology -- both hardware and software -- towards this goal of teraflops."

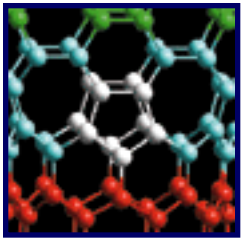
Feiereisen, along with other speakers, voiced concern about hitting the upper limits of the capabilities of the silicon chip -- called the "silicon barrier" -- and the related device-size barrier. Feiereisen said that the solution to these limits might be in distributed computing, noting that this brings its own challenges. "The other difficulty we have in getting beyond this brick wall is the ability to manage large numbers of processors," he said.

Feiereisen wants to form collaborative relationships with other organizations. "This program sees us being able to get to petaflops with the extension of the kinds of technology we have now. But we're very much interested in funding and supporting device modeling research and system software research to be able to get us beyond that brick wall."

Expertise in Process and Device Modeling

Meyya Meyyapan, device modeling integrated project team manager, discussed NASA's capabilities in process and device modeling. According to Meyyapan, Ames is set up to conduct research in computational device physics, computational and experimental nanotechnology, computational chemistry, and process modeling.

"All these tasks are very well suited to NASA Ames' traditional resources and talents," he said. Meyyapan finished his talk by expressing interest in collaborative work. "We hope that the industry will be the end users of the Ames-developed expertise."



Nanoscale metal-semiconductor-metal contact devices

Device Modeling at Ames

Subhash Saini, workshop chair and lead of the newly formed [modeling and simulation group](#) in the NAS Systems Division, discussed the larger picture of device modeling at Ames. "[Ames has] experience in large-scale modeling and simulation both in computational fluid dynamics as well as in chemistry," which gives the center a natural advantage in device modeling research, he said. Saini added that the center has many other advantages, including leading-edge high-performance computer systems and a Silicon Valley location, with Stanford University and the University of California at Berkeley nearby.

In addition, Saini emphasized Ames' experience with other kinds of modeling that translate to device modeling. "Here at Ames we have the technology of creating very complicated unstructured grids, and we are very good at scientific visualization and virtual reality."

Keynote Echoed 'Collaborative Spirit'

In the keynote address, Carl Kukkonen, director of the Center for Space Microelectronics at NASA's [Jet Propulsion Laboratory \(JPL\)](#), discussed projects being done in his division, some of the quandaries they face, and in the spirit of collaboration, asked for solutions from other participants.

JPL is funded by NASA and is also a division of Caltech, in Pasadena. Kukkonen explained that while the laboratory is attached to the university, "We're a mission agency. We don't do very much basic research, we do mission-oriented applied research.

"There are two things we don't do: We do not do integrated circuits -- industry does integrated circuits very well and we contract for them. We don't do basic research. We rely on the universities to do that. When we see something interesting in the universities that we need for our missions, we try to grab it and we'll give a contract to a particular investigator," he said.

Budgetary Concerns Fuel Challenges

This limited role is in part due to of budgetary concerns. Kukkonen explained that "in today's more frugal environment, Congress has said, 'we're not going to authorize any more billion-dollar operations at one time, and if you want to continue doing space science you're going to have to figure out ways to do this faster, better, and cheaper.'" JPL is trying to "figure out how we can do the same science in the less-expensive mode," he said.

For example, the way NASA conducts its space missions can be changed, Kukkonen said. "When you do [a mission] once a decade, we tend to freeze the technology seven years in advance of flight." In doing so, "you're four or five generations behind before you ever launch. If it takes five years to get there, you're another four generations behind and you get there with ancient technology."

The more budget-conscious approach, according to Kukkonen, is to break missions up into several smaller missions. For example, the [Cassini mission](#) (launched October 15) to gather data from Saturn, requires twenty instruments on a probe. "So ideally, we'd break up those twenty instruments and...launch five spacecraft, once every two years. If you launch them every two years you can insert new technology more often, and if one blows up you only lose twenty percent of the mission."



Best Investment: Miniature Instruments

Another plan of attack is miniaturization of instruments. "The best investment we can make is to try to minimize the scientific instruments and subsystems on the spacecraft," said Kukkonen. To illustrate his point, Kukkonen showed [a chart](#) of decreasing spacecraft sizes and the budgets for those spacecraft. "We're going from big spacecraft to small spacecraft. The Mars Pathfinder mission is a step along the way," he explained.

According to Kukkonen, "Pathfinder landed in one little spot on Mars [July 4], and the rover so far has progressed a grand total of ten meters from the spacecraft." While acknowledging that this mission has supplied a lot of data, Kukkonen likened it to a Martian spacecraft landing in the Sahara Desert. "Although we're getting lots of data about Mars, we're getting data about one very specific location. What you'd really like to do is get global

The decreasing size of
NASA missions



Mars microprobe

coverage, so you'd like to drop probes all over the planet." What Kukkonen has in mind is a [probe](#) about a foot long and four inches in diameter, which can be dropped in large quantities over the surface of a planet to gather data and perform tests.

Future Spacecraft Like 'Ancient Mariners'

While NASA tries to cut costs of missions and improve technological timeliness, there is also a need to cover new territory. "Our sensors and detectors are getting so good now that we need high-performance computing on the spacecraft to handle all those large forms of data. And in order to reduce operations cost, we want to have spacecraft autonomy." Rather than taking commands from mission control, the spacecraft "will navigate like the ancient mariners, by looking at the stars," he said, adding that "we also need high-performance computing on the ground to turn that information into knowledge."

Reiterating the need for collaboration, Kukkonen said that the kind of modeling JPL needs is "not only electronic device modeling, but mechanical modeling.... We're doing some crude things, and we could use a lot of help."

To order the workshop proceedings, send email to doc-center@nas.nasa.gov.



Updated: August 3, 1999
WebWork: [Publications Media Group](#)
NASA Responsible Official: [Bill Feiereisen](#)



QUESTIONS



HELP



In This Issue...

[SC97 Special](#)
[Section: NASA](#)
[Centers](#)



[Collaborate on Demos](#)

[Origin2000 Update](#)

[Kutler Focuses on IPG](#)

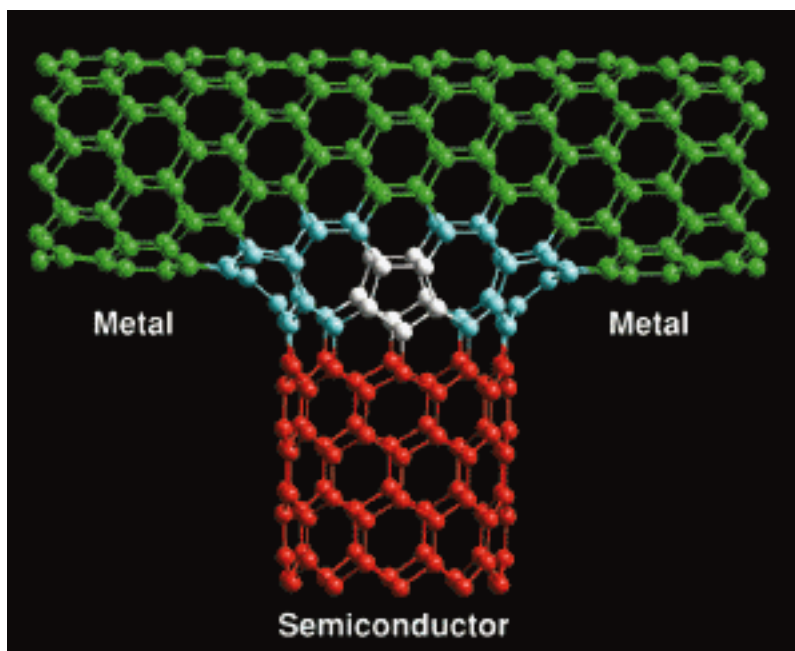
[CFD Visualization Library](#)

Device Modeling
 Workshop

[Contiguous Improvement](#)

[Whitney Team](#)

[New Approach to Mass](#)
[Storage](#)



Nanoscale metal-semiconductor-metal contact devices, made of carbon nanotube "T-junctions," from "Carbon Nanotube Junctions as Building Blocks for Nanoscale Electronic Devices," by Madhu Menon, Deepak Srivastava, and Subhash Saini. T-junctions provide a challenge to the conventional rules applicable to tube bends because both sides of the joint are topologically equivalent. As a result, more heptagons than pentagons can be expected at the junction. An alternative route to the formation of T-junctions is not constrained by the usual heptagon-pentagon defect pair considerations. In particular, this figure shows a metal-semiconductor-metal T-junction ((5,5)-(10,0)-(5,5)), composed of 314 atoms. The numbers chosen are large enough to avoid the effects of dangling bonds at the edges on the junction. The (10,0) tube is semiconducting, and the (5,5) tube is a semi-metal.

[Return To Main Article](#)



In This Issue...

[SC97 Special](#)
[Section: NASA](#)
[Centers](#)



[Collaborate on Demos](#)

[Origin2000 Update](#)

[Kutler Focuses on IPG](#)

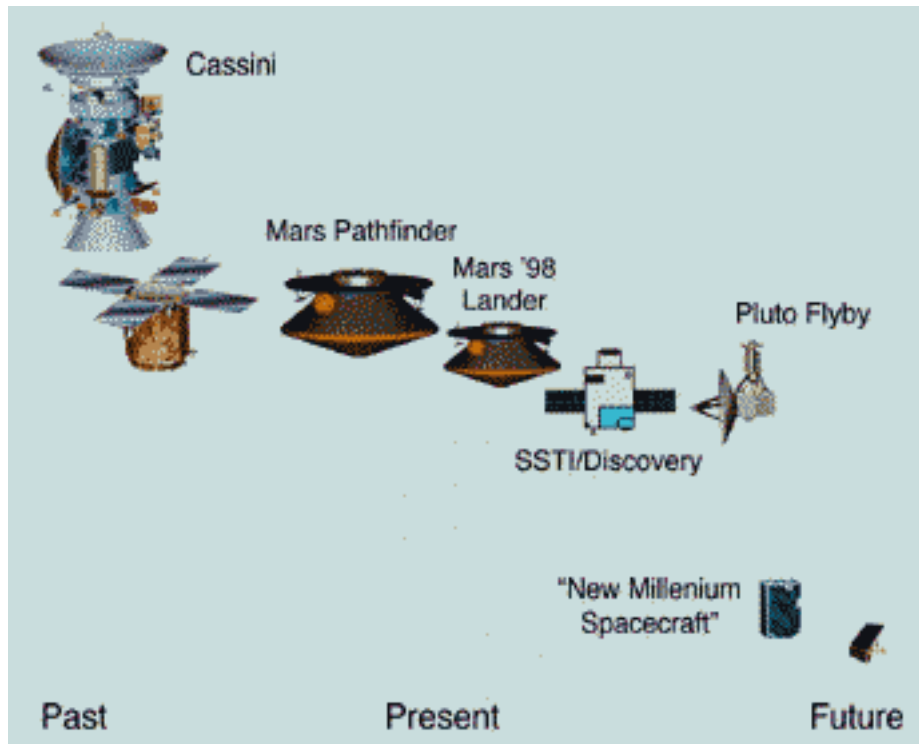
[CFD Visualization Library](#)

[Device Modeling](#)
[Workshop](#)

[Contiguous Improvement](#)

[Whitney Team](#)

[New Approach to Mass](#)
[Storage](#)



The decreasing size of NASA missions is illustrated in this slide from Carl Kukkonen's keynote address at the Second NASA Device Modeling Workshop. With the help of device modeling, Kukkonen predicted, future spacecraft will be much smaller, and can be launched more often and at a lower cost to taxpayers. One of the smallest crafts mentioned in his talks is the [Mars microprobe](#).

[Return To Main Article](#)



In This Issue...

[SC97 Special](#)
[Section: NASA](#)
[Centers](#)
[Collaborate on Demos](#)



[Origin2000 Update](#)

[Kutler Focuses on IPG](#)

[CFD Visualization](#)
[Library](#)

[Device Modeling](#)
[Workshop](#)

[Contiguous Improvement](#)

[Whitney Team](#)

[New Approach to Mass](#)
[Storage](#)

'Contiguous Improvement' in Job Performance

by [Daniel DePauk](#) and [John Hardman](#)

Second in a series of articles designed to help HSP users maximize performance from the Cray systems by making small but significant changes in the way their programs are run. Last issue, we discussed [using the \\$BIGDIR directory for batch jobs](#) (via the Session Reservable File System) instead of slower "home" filesystems to reduce CPU time by as much as 34 percent. In this issue, the focus is on further increasing disk I/O performance by using contiguous file system disk blocks, where possible.

Since disk access is one of the most time-consuming activities involved in program execution, it makes sense to look carefully in this area for efficiencies that can be realized at runtime. Fortunately, the Cray UNICOS operating system has several features that allow users to override the default allocation of disk space for their jobs to achieve better performance. Effective use of these features can result in a significant speed-up in the processing of I/O requests through the organization of disk allocations into contiguous blocks.

Within this article...

[Advantages of "Read Ahead, Write Behind"](#)

[How to Obtain](#)
[Contiguous Blocks](#)

[cp Command](#)

[setf Command](#)

[assign Command](#)

[Significant](#)
[Performance](#)
[Improvements](#)

Before discussing the techniques for creating contiguous block allocations, let's take a moment to look at the performance penalties imposed by the default behavior of the system. Unless you provide the system with the allocation size for a new file, that file is created with a size of one disk block (the actual size of the block varies, depending on the type of disk device). The filesystem block selected by the operating system is normally the first free block.

As a program writes to the file and fills the file's current allocation, a system request is made for another block. The requested block will, again, be the next available block of the file system. This block will rarely be an adjacent block to the first one, because of the intervening system activity of other users. So, under normal conditions, a large file written to the system using the default method will consist of a series of single, non-adjacent blocks. This increases the processing time for I/O requests because the device must be accessed again and again to read or write to all the non-adjacent blocks.

Advantages of 'Read Ahead, Write Behind'

A Cray UNICOS system disk I/O feature known as "Read Ahead, Write Behind" is responsible for most of the speed-up available using contiguous blocks. When a program reads from a file using system-buffered I/O, the full filesystem block containing that data is retrieved from the disk and is placed into a system buffer. The Read Ahead, Write Behind feature will then fill other system buffers with up to seven more blocks if they are contiguous in the filesystem. Also, when writing a file, the data from system buffers can be moved to disk, as required, in up to eight contiguous blocks at a time.

Even if a program has been set up to use direct I/O instead of the default system buffers, the use of contiguous blocks may help. With the Read Ahead, Write Behind feature, even if the program requests direct I/O, the system may use buffered I/O, if it determines that it will improve overall performance.

The benefits of using contiguous blocks accumulate quickly as the file size increases. Let's say you're running a program which reads a file that is 800 filesystem blocks in size. If the file is completely contiguous, the Read Ahead feature can be exploited, and the program will only have to wait for disk access one out of eight times that a read is requested from

the next block. This means that the disk would be accessed only 100 times in the course of reading the entire file. In contrast, the same file, read from completely non-adjacent blocks, would require 800 disk accesses. Similar performance increases can be obtained when programs write data to disk.

How To Obtain Contiguous Blocks

A number of different techniques can be used to create contiguous block allocations. Three of the most useful techniques are described briefly here. Since there are several options for these commands, users are encouraged to check the system documentation for more details, or to contact the NAS scientific consultants for assistance.

Operating system activity is roughly the same for file allocation requests made with each of the techniques described below. The system attempts to allocate blocks in one contiguous space in the filesystem. If this isn't possible, the largest available space is given to the file, then the next largest, and so on, until the request has been fully allocated. In most cases, the file will be made of large enough parts to allow the Read Ahead, Write Behind feature to effectively improve program performance.

***cp* Command**

When a file is to be read by a program during execution, simply using the *cp* command to copy the file to \$BIGDIR first will allow you take advantage of contiguous block allocation. The system automatically calculates the contiguous space required in \$BIGDIR before beginning the data transfer. This feature enhances the previously discussed [advantages of using \\$BIGDIR](#) when running your jobs.

***setf* Command**

When a file is created by a program, the *setf* command can be used to initialize filesystem space for that file. This command will also lengthen the file space allocation if the file already exists. The command option *-n size* allocates the space. Refer to the man page on one of the HSP systems for complete information.

***assign* Command**

When working with the Fortran run-time I/O libraries, the *assign* command also allocates filesystem space when a file is opened by a Fortran program. The command option *-n sz* allocates space. Refer to the man page for *assign* on one of the HSP systems for more information.

Significant Performance Improvements

These methods for sharply cutting disk I/O overhead for your jobs can yield a potential 87 percent reduction in disk I/O wait time, with obvious benefits for overall program performance. Fortunately, taking advantage of these benefits is straightforward and requires little or no modification of your code.

For more information on these methods, contact the NAS scientific consultants at (650) 604-4444 or (1-800) 331-8737, or send email to nashelp@nas.nasa.gov.

If you've found these suggestions for "contiguous improvement" to be helpful, please [email](#) your comments.



In This Issue...

[SC97 Special
Section: NASA
Centers
Collaborate on Demos](#)



[Origin2000 Update](#)

[Kutler Focuses on IPG](#)

[CFD Visualization
Library](#)

[Device Modeling
Workshop](#)

[Contiguous Improvement](#)

[Whitney Team](#)

[New Approach to Mass
Storage](#)

'Whitney' Team Weighs Cluster Network Options

Samuel A Fineberg and [Jeffrey C. Becker](#)

Recent developments in "commodity" network, disk, and CPU technology have raised the performance of a \$2,500 PC to the level that a \$25,000 workstation was at one or two years ago. This presents an opportunity to pursue a shortcut to petaflops computing -- performance that's a thousand times faster than today's fastest supercomputer.

In April, members of the NAS Systems Division's [parallel group](#) installed a 30-node cluster of personal computers (PCs), referred to as the "Whitney" prototype (*see [NAS News, March/April '97](#)*.) Using clustering technology, the team's goal is to build a large testbed with supercomputer-class performance for less than one-tenth the cost of a supercomputer-class system.

The Whitney project, aimed at designing a large-scale testbed from inexpensive "commodity" parts such as those found in PCs, has been under way for about seven months. The project team is currently evaluating system component options, including those to determine which network technologies will be best suited for the scientific codes run at the NAS Facility.

40-Node Prototype Will Grow to 500

The prototype's initial 30-node configuration installed last spring has been increased to 40 nodes, with plans to expand to 500 nodes over the next year. The prototype is constructed of 200-megahertz (MHz) Intel Pentium Pro-based PCs. Each PC runs an individual copy of the Linux

Within this article...

[40-Node Prototype
Will Grow to 500](#)

[Performance Test
Results](#)

[Production Planned
for Fall '98](#)

operating system. A front-end system acts as the user, file, and compile server. In addition to running MPIRUN for job startup, the prototype runs PBS (the NAS-developed Portable Batch System) as a batch scheduler.

The prototype also uses the Portland Group's Fortran 77 and High Performance Fortran compilers, which produce significantly faster code than the public-domain Fortran compiler, called "g77."

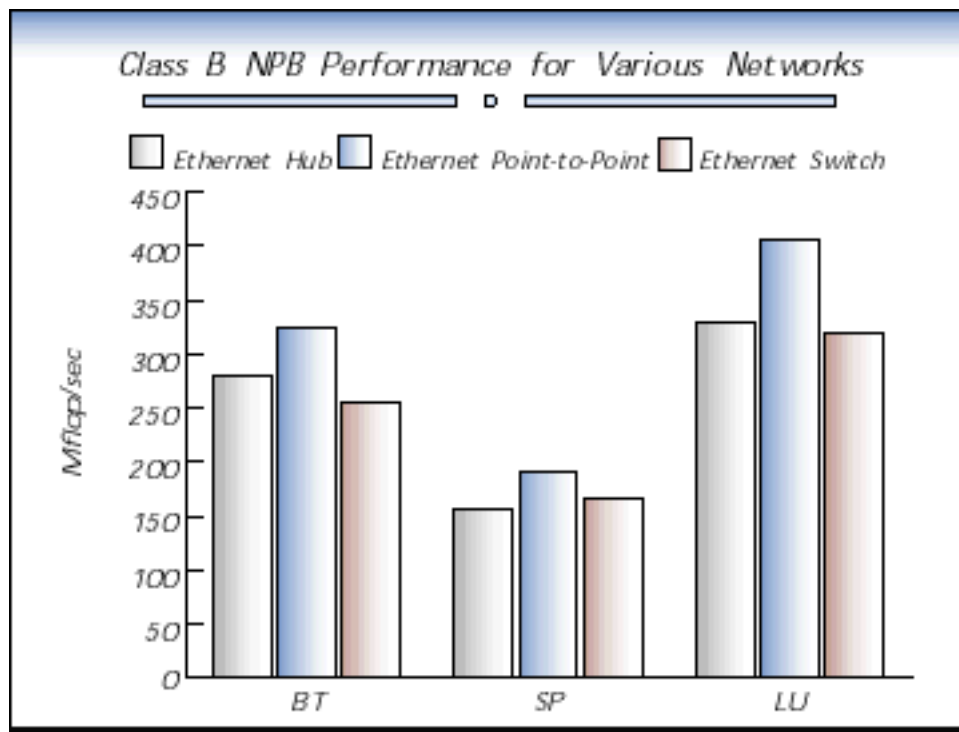
The team is now in the process of testing options for interconnecting the Whitney nodes. Two options include a 100-million-bits-per-second Ethernet (Fast Ethernet), as well as a billion-bits-per-second network called Myrinet. These network options, as well as the network topology itself, range widely in cost and performance.

Cost is a key factor in differentiating each of these networking options. Fast Ethernet adapters cost about \$50 each, hubs cost about \$75 per port, and switches cost about \$200-\$700 per port. Unfortunately, to achieve the scalability needed to build a 500-processor machine, the design team cannot rely exclusively on hubs, and even the largest switches cannot scale to 500 ports.

Myrinet adapters cost approximately \$1400 per node, and with the switching hardware necessary to build a mesh, the cost goes up to about \$1800 per node. Although this would nearly double the cost of each PC in the Whitney system, Myrinet has over 10 times the bandwidth of the other choices -- 98 megabytes per second (MB/sec) versus 8 MB/sec for Fast Ethernet -- and one-tenth the latency -- 20 microseconds versus Fast Ethernet's 180 microseconds.

Performance Test Results

The options investigated include Fast Ethernet attached with switches, hubs, point-to-point links, and combinations of these. Myrinet testing has been concentrated on a two-dimensional mesh of switches. To evaluate the performance of each of these networks, we measured latency and bandwidth, and employed the NAS Parallel Benchmarks (NPB). The graph shows some Fast Ethernet results.



The NPB results obtained in October indicate that Myrinet's performance outshines Fast Ethernet, although the ratio of performance to price is worse. However, Myrinet provides better scalability than Fast Ethernet, so it may be better for large systems. In addition, the hub-based topologies of Fast Ethernet are far less scalable than switch-based or routed topologies, even though the latency on the hub networks is lower.

Packaging to Meet Space Requirements

Another key issue the team is looking at is packaging. Currently, the 40 nodes are spread out on five tables and on the floor. A 500-node system will require a much denser packaging solution. Standard system racks seem like an obvious approach, but rack-mounted PC cases are larger than the mini-towers used in the prototype -- and cost ten times as much. As another solution, we are looking at shelving systems that will house a 500-node system in about 300 square feet of floor space.

Production Planned for Fall '98

While evaluation of components continues, the Whitney project team has also begun software development. During 1998, in addition to expanding the system to 500 processors, the team will install and run the parallel-aware version of PBS, and will continue developing a parallel file system. The testbed is expected to go into production

sometime next fall. Over the next three years, the system will expand to 5,000 processors in order to examine the effects of massive parallelism on systems software.

For the latest details on the Whitney project, see [the Whitney Project home page.](#)



In This Issue...

[SC97 Special](#)
[Section: NASA](#)
[Centers](#)
[Collaborate on Demos](#)



[Origin2000 Update](#)

[Kutler Focuses on IPG](#)

[CFD Visualization](#)
[Library](#)

[Device Modeling](#)
[Workshop](#)

[Contiguous Improvement](#)

[Whitney Team](#)

[New Approach to Mass](#)
[Storage](#)

New Approach to Mass Storage: Faster, Better, Cheaper and Lazier

by [John Lekashman](#)

The NAS Systems Division has recently embarked on a significantly new path for handling mass storage. A basic goal of this development effort is to build systems that have very large capacity and high performance, as well as the advantages of commodity products.

What is commodity mass storage, anyway, and why would anyone want to try it? Commodity mass storage is just what it sounds like. Buying commercial off-the-shelf hardware and software--including control processors, disk and tape storage units, and network components--from multiple manufacturers, as opposed to ordering custom work from a particular vendor.

Within this article...

[Mass Storage 'on the Cheap'](#)

[Philosophy: No](#)
[Hostages](#)

[System Design Must](#)
[Scale](#)

[Just Add Water and](#)
[Stir](#)

Mass Storage 'on the Cheap'

The basic motivations for this experiment--it's unclear at this point whether it will actually work as planned--is simple:

- We're cheap.
- We're lazy.

Well, okay, we're not really *cheap*--we prefer to think of it as being smart shoppers. Paying exorbitant overhead for mass storage isn't high on our list. Mass storage systems at supercomputer centers such as the NAS Facility run upwards of \$18 million--the current NAS system cost an initial \$11 million, with the remainder going toward upgrades. The

new system we're building, which is on par with such facilities (scaled to current requirements), is expected to cost \$3 million initially, with a few more million thrown in over the years if we need to expand system capacity.

And we're only lazy about some things. Like spending lots of time negotiating with vendors over how or when--if ever--their software will work for us. Or what the licensing charge per byte stored will be.

Philosophy: No Hostages

The fundamental philosophy and requirements behind building the NAS Facility's commodity system are:

- Our data is ours. No one can hold it hostage.
- System design must scale and be able to operate over a wide dynamic range.
- Turnaround on system capacity changes must be able to occur instantly.

One of the first realizations when building a commodity system is the key role of hardware and software interfaces. These interfaces have to be "wide open"--that is, they can't be dependent on proprietary products and have to be available from a variety of sources. This means that novendor-specific implementation can exist in the system across an interfaceboundary. And the entire piece that a vendor provides must conform to the interfaces defined for the other pieces. Standard technologies such as TCP/IP, RAID, SCSI, Fibre Channel, OpenVault, and DMAPI are key--and must be the total underpinning of the system.

In this scenario, it's perfectly fine for the vendor of a host platform that accesses our data to have its own native, proprietary filesystem. It's *not* okay for our data to be stored in that filesystem in a way that makes reading the data out of that filesystem cumbersome--it shouldn't take more than a day or so.

The benefits of this approach are many: No vendor-proprietary filesystems, no charges per byte stored. And if things don't work out, we can change our minds in short order about using a particular vendor. Vendors need to know that they can be replaced if performance or service is lacking.

System Design Must Scale

The requirement that system design must scale and operate over a wide dynamic range means that the system must be able to be large or small. The storage capacity must be able to change--on demand and with little effort--whether dealing with 10 terabytes of data or 10,000 terabytes

The idea is to be able to place an order for 10 more silos, or 10 more terabytes of disk, roll it into the working system, and continue without a hiccup. Or, to be able to drop in a small system quickly, if needed.

A key part of this requirement is that all common components be available from multiple sources; interoperability standards can't be built on one vendor's product. If supplies come from just one vendor and that vendor can't deliver, then we're up the proverbial creek without a paddle.

As an example, one of the NAS Facility's previous storage systems was blocked on capacity for nine months because the vendor didn't develop I/O bays to attach to the control processors. Parts must interoperate, or they won't be considered for inclusion in our system. Only in this way can we continue to scale.

Just Add Water and Stir

Turnaround time on system capacity changes must be able to occur instantly. It's vital to be able to purchase extra storage capacity as needed, and "turn it off" as needed--without complex licensing, without months of negotiation, without all sorts of special work. We want to be able to literally go down to one of our local computer stores and buy it, or order it from any number of places on the Web and charge it on a credit card.

Financial management considerations play a large role in the success of this requirement. As part of this process, we established that no individual piece of the whole will cost more than about \$300,000--or, about 10 percent of the whole budget. This way, if individual parts or vendors completely fail, they can be tossed, and we can absorb the \$300,000 loss. Other sites will establish their own parameters, depending on the scale of the system deployment and the facility.

In addition, the idea of a multi-year lease of equipment--using a major portion of our budget to pay the lease--was thrown out the window. Because we determined that our data won't be held hostage, we can't be in a position to owe money on the system. It's hard to buy a new car if you still owe more money on the one you have than it's worth.

For more information on the commodity mass storage project, send email to lekash@nas.nasa.gov. Watch for updates in future issues of NAS News.

**In this issue...**

[SC97 Special](#)
[Section: NASA](#)
[Centers](#)



[Collaborate on Demos](#)

[Origin2000 Update](#)

[Kutler Focuses on IPG](#)

[CFD Visualization Library](#)

[Device Modeling](#)
[Workshop](#)

[Contiguous Improvement](#)

[Whitney Team](#)

[New Approach to Mass](#)
[Storage](#)



Credits

Executive Editor: Thomas Lasinski

Editor: Jill Dunbar

Staff Writer: Ayse Sercan

Contributing Writers: Jeffrey C. Becker, Daniel DePauk, Thomas W. Faulkner, Samuel A. Fineberg, John Hardman, Chris Henze, John Lekashman, Patrick Moran

Image Coordinator: Joel Antipuesto, Chris Gong

Online Page Layout and Graphics: Joel Antipuesto, Chris Gong, Rosemary Wadden

Other Contributors: Richard Anderson, Asen Asenov, Robert Biedron, Timothy Brice, Steve Bryson, James Donald, David Kenwright, Paul Kutler, Chris Kuzmal, Art Lazanoff, Marcia Redmond, Lisa Reid, Ian Stockdale, Pichuraman Sundaram, Jim Taft, Veer Vatsa, Stanley Williams

Editorial Board: Cristy Brickell, Jill Dunbar, Chris Gong, Thomas Lasinski, Nateri Madavan, Patrick Moran, George Myers, Ayse Sercan, Harry Waddell